UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/676,658	09/30/2003	Zhenhai Lin	50277-2280	8991

42425	7590	08/08/2006

HICKMAN PALERMO TRUONG & BECKER/ORACLE
2055 GATEWAY PLACE
SUITE 550
SAN JOSE, CA 95110-1089

EXAMINER
VAUTROT, DENNIS L

ART UNIT	PAPER NUMBER
2167	

DATE MAILED: 08/08/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

	Application No.	Applicant(s)
Office Action Summary	10/676,658	LIN, ZHENHAI
	Examiner	Art Unit	
	Dennis L. Vautrot	2167	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE <u>3</u> MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *30 September 2003*.

2a)☐ This action is **FINAL**.   2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-52* is/are pending in the application.

    4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-52* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *30 September 2003* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

    Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

    Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

    a)☐ All   b)☐ Some * c)☐ None of:

      1.☐ Certified copies of the priority documents have been received.

      2.☐ Certified copies of the priority documents have been received in Application No. _____.

      3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

    * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1)☒ Notice of References Cited (PTO-892)

2)☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3)☒ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
    Paper No(s)/Mail Date *5/20/2004*.

4)☐ Interview Summary (PTO-413)
    Paper No(s)/Mail Date. _____.

5)☐ Notice of Informal Patent Application (PTO-152)

6)☐ Other: _____.

DETAILED ACTION

Information Disclosure Statement

1.   The information disclosure statement (IDS) submitted on 20 May 2004 has been

received and entered into the record. Since the IDS complies with the provisions of

MPEP § 609, the references cited therein have been considered by the examiner. See

attached forms PTO-1449.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of
> matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the
> conditions and requirements of this title.

2.   Claims 24-52 are rejected under 35 U.S.C. 101 because the claimed invention is

directed to non-statutory subject matter. The storage medium, as defined in the

specification in paragraphs [0069] through [0071] includes acoustic and light waves.

This is not tangibly embodied in a computer-readable medium, and hence non-statutory.

Claim Rejections - 35 USC § 102

3.   The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that

form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by
> another filed in the United States before the invention by the applicant for patent or (2) a patent
> granted on an application for patent by another filed in the United States before the invention by the
> applicant for patent, except that an international application filed under the treaty defined in section
> 351(a) shall have the effects for purposes of this subsection of an application filed in the United States

only if the international application designated the United States and was published under Article 21(2)
of such treaty in the English language.

4.   Claims 1 –2, 4 – 14, 16 – 31, 33 – 43, and 45 – 52 are rejected under 35

U.S.C. 102(e) as being anticipated by **Menon** (6,615,204).

5.   Regarding Claims 1, 24, and 30, **Menon** teaches a method for storing data in a

repository comprising the steps of: storing in a first table data for one or more default

attributes of a first object type used by an application. (See fixed mapping asset table A

FIG. 12, 1230); storing in a second table, separate from said first table, data for one or

more default attributes of a second object type used by said application. (See fixed

mapping asset table B FIG. 12, 1231); and storing in a third table, separate from said

first and second tables, data for a first custom attribute of said first object type and data

for a second custom attribute of said second object type, wherein said first custom

attribute and said second custom attribute have the same data type (See FIG. 12, item

1106, where custom attributes are stored by the same type).

6.   Regarding claim 2, 26, and 31, **Menon** additionally teaches said third table

includes at least one instance-identifying column (See Figure 1103, which represents

the third table referred to in the claim.) wherein each row of said third table stores in

said at least one instance-identifying column data that uniquely identifies an object

instance that is associated with the row (See column 20, lines 40-42 "Note that the

same entry also comprises an object-id which maps back to the asset represented by

the entry in the asset table." The object id represents the instance-identifying column.);

at least one attribute-identifying column, wherein each row of said third table stores in

said at least one attribute-identifying column data that identifies a custom attribute of the

object instance that is associated with the row; and at least one value column, wherein

each row of said third table stores in said at least one value column one or more values

for the custom attribute that is identified in said at least one attribute-identifying column;

and said at least one value column of said third table stores data that has the same data

type as said first custom attribute and said second custom attribute. (See column 20,

lines 37-40 "For example, as depicted by the arrow 1114, the 'String' attribute (name

and value) associated with the entry 1104a is stored in the String metadata table 1106a

in entry 1108a." This shows the table storing the attribute and its value respectively.)

7.   Regarding claims 4, and 33, **Menon** additionally teaches the step of retrieving an

object instance of said first object type, wherein said object instance of said first object

type includes data from said third table associated with said first custom attribute of said

first object type. (See column 32, lines 55-59 "In step 1406, database operations are

supported by relating objects stored in the fixed mapped table to their extensions in the

flexible mapped table. The extensions are related to their respective objects via the

object ID field in each entry." The data associated with the firs attribute in the case is

the object ID field.)

8.   Regarding claims 5 and 34, **Menon** additionally teaches said third table stores values for custom attributes of a plurality of object types including said first object type and said second object type (See FIG 11, item 1102); and the method further comprises assigning to every instance of every object type of said plurality of object types an instance identifier value that is unique relative to every instance of every object type of said plurality of object types (See FIG 11, item 1103).

9.   Regarding claims 6, 18, 27, 35, and 47 **Menon** additionally teaches the step of storing, in a catalog table, data defining said first custom attribute of said first object type and said second custom attribute of said second object type (See FIG. 11, showing the custom attribute data being stored.)

10.   Regarding claims 7, 19, 28, 36, and 48, **Menon** additionally teaches said catalog table includes: at least one first catalog column, wherein each row of said catalog table stores, within said at least one first catalog column, data that identifies an object type associated with the row (See column 20, lines 9 – 13 "Each box depicted within the second column 1105 represents a particular attribute for the associated asset [object]. Each attribute typically comprises three elements: (1) an attribute name; (2) an attribute value; and (3) an attribute type." Here the type is part of the attribute field.),

at least one second catalog column, wherein each row of said catalog table stores, within said at least one second catalog column, data that identifies a custom attribute of the object type that is associated with the row (See column 20, lines 19-20

"Note that each entry can have a variable number of attributes within the second

column."), and

at least one third catalog column, wherein each row of said catalog table stores,

within said at least one third catalog column, data identifying a data type of the custom

attribute that is identified in said second catalog column (See column 20, lines 11 – 13

"Each attribute typically comprises three elements: (1) an attribute name; (2) an attribute

value; and (3) an attribute type." Here the type is also part of the attribute field.)

11.   Regarding claims 8 and 37, **Menon** teaches the step of retrieving, in response to

a request for an object instance of said first object type, the value of said first custom

attribute associated with said object instance (See column 20, lines 9 – 13 "Each box

depicted within the second column 1105 represents a particular attribute for the

associated asset [object]. Each attribute typically comprises three elements: (1) an

attribute name; (2) an attribute value; and (3) an attribute type."); by performing the

steps of: determining the data type of said first custom attribute from said third catalog

column of a catalog-table row stored in said catalog table (See column 20, lines 9 – 13

"Each box depicted within the second column 1105 represents a particular attribute for

the associated asset [object]. Each attribute typically comprises three elements: (1) an

attribute name; (2) an attribute value; and (3) an attribute type." And lines 14-16 "The

type is selected from a predefined list of types. As will be described below, each

predefined type is associated with one of the separate metadata tables." The type is

also stored in the attribute field.); wherein: data in said at least one first catalog column

of said catalog-table row matches data identifying said first object type (See FIG. 11,

item 1114); and data in said at least one second catalog column of said catalog-table

row matches data identifying said first custom attribute (See FIG. 11, item 1116); based

on the data type of said first custom attribute, determining the identity of said third table

(See column 20, lines 14-16 "As will be described below, each predefined type is

associated with one of the separate metadata tales."); and retrieving, from a value

column of a third-table row stored in said third table, the value of said first custom

attribute (See column 20, lines 11-13 "Each attribute typically comprises three elements:

(1) an attribute name; (2) an attribute value; and (3) an attribute type."), wherein: data

uniquely identifying said object instance matches data in at least one instance-

identifying column of said third-table row (See FIG. 11, item 1103); and data identifying

said first custom attribute matches data in at least one attribute-identifying column of

said third-table row (See FIG. 11, item 1103, where the object id matches).

12.   Regarding claims 9, 29, and 38, **Menon** additionally teaches the step of storing in

said catalog table data defining a custom object type, separate from said first object

type and said second object type, wherein the step of storing said custom object type

includes the step of inserting a row into said catalog table (See column 30, lines 43-45

"In this embodiment, the extensions (e.g., object B' 1202b) are stored using asset table

1102 of the flexible mapped portion."), wherein said row includes: within said at least

one first catalog column, data that identifies said custom object type (See column 20,

lines 11-13, specifically (1) name); within said at least one second catalog column, data

that identifies a custom attribute of said custom object type (See column 20, lines 1 and

2, specifically object id), and within said at least one third catalog column, data

identifying a data type of said custom attribute that is identified in said at least one

second catalog column (See column 20, lines 11-13, specifically (3) type).

13.   Regarding claims 10, 22, 39, and 51, **Menon** additionally teaches said catalog

table stores data that identifies custom attributes for a plurality of object types (See FIG.

11, item 1105, note different types); the method further comprises performing the

following steps in response to said application being launched: reading said catalog

table to determine custom attributes from said plurality of object types (See column 17,

lines 7-9 "For example, such tool 224 can be used to read or modify attribute values

and/or to read an asset directly."); and based on the information from said catalog

table, constructing in volatile memory data structures that indicate the custom attributes

of each of said plurality of object types (See column 27, lines 2-5 "Once in memory, a

client uses accessor methods of AmsBase to get individual attributes of a data object.

Contents may be checked out into memory or into memory or into a file in the local file

system."); and in response to a request to access an object instance of a particular

object type of said plurality of object types, inspecting said data structures, without

accessing said catalog table, to determine the custom attributes of said particular object

type (See column 27, lines 28-32 "Thus, an application program could at runtime, query

this property list to determine the structure, i.e. attributes and types, and values of the

object's metadata. Consequently, It is not necessary for the application to use built-in

accessor methods to read the attributes.")

14.   Regarding claim 11, 23, and 40, **Menon** additionally teaches said catalog table

stores data that identifies custom attributes for at least one object type (See FIG. 11,

item 1105, note different types); the method further comprises performing the following

steps in response to a request to access an object instance of a particular object type:

reading said catalog table to determine the custom attributes of said particular object

type (See column 17, lines 7-9 "For example, such tool 224 can be used to read or

modify attribute values and/or to read an asset directly."); and based on the information

from said catalog table, constructing in volatile memory data structures that indicate the

custom attributes of said particular object type (See column 27, lines 2-5 "Once in

memory, a client uses accessor methods of AmsBase to get individual attributes of a

data object. Contents may be checked out into memory or into memory or into a file in

the local file system."); and in response to a subsequent request to access an object

instance of said particular object type, inspecting said data structures, without accessing

said catalog table, to determine the custom attributes of said particular object type (See

column 27, lines 28-32 "Thus, an application program could at runtime, query this

property list to determine the structure, i.e. attributes and types, and values of the

object's metadata. Consequently, It is not necessary for the application to use built-in

accessor methods to read the attributes.")

15.   Regarding claims 12 and 41, **Menon** teaches a method for storing data in a

repository comprising the steps of: storing in a first table data for one or more default

attributes of an object type used by an application (See figure 12, asset tables A and B,

which store the default attributes); storing in a second table, separate from said first

table, data for a first custom attribute of said object type that is of a first data type (See

Figure 11, item 1106a and Figure 12, item 1220, showing the different data type being

stored in a separate table.); and storing in a third table, separate from said first and

second tables, data for a second custom attribute of said object type that is of a second

data type, wherein said first custom attribute and said second custom attribute have

different data types (See Figure 11, Item 1106b and Figure 12, item 1220 showing the

different data type being stored in a separate table.)

16.   Regarding claims 13 and 42, **Menon** additionally teaches said object type is a

first object type of a plurality of object types used by said application (See figure 11,

item 1104 where they show the various objects of different object types); and the

method further includes: storing in a fourth table, separate from said first, second and

third tables, data for one or more default attributes of a second object type of said

plurality of object types (See figure 12, item 1231, showing the asset table B, storing

default attributes); and storing in said second table data for a third custom attribute of

said second object type, wherein said third custom attribute of said second object type

is of said first data type (See figure 12, item 1106, storing the data types of custom

attributes separately).

17.   Regarding claim 14 and 43, **Menon** additionally teaches said second table

includes at least one instance-identifying column, wherein each row of said third table

stores in said at least one instance-identifying column data that uniquely identifies an

object instance that is associated with the row (See column 20, lines 40-42 "Note that

the same entry also comprises an object-id which maps back to the asset represented

by the entry in the asset table." The second table here is essentially the third table from

claim 2. Both refer to the custom attribute tables.);

at least one attribute-identifying column, wherein each row of said third table

stores in said at least one attribute-identifying column data that identifies a custom

attribute of the object instance that is associated with the row; and at least one value

column, wherein each row of said second table stores in said at least one value column

one or more values for the custom attribute that is identified in said at least one

attribute-identifying column; and said at least one value column of said second table

stores data that has the same data type as said first custom attribute and said second

custom attribute. (See column 20, lines 37-40 "For example, as depicted by the arrow

1114, the 'String' attribute (name and value) associated with the entry 1104a is stored in

the String metadata table 1106a in entry 1108a." This shows the table storing the

attribute and its value respectively.)

18.   Regarding claims 16 and 45, **Menon** additionally teaches the step of retrieving

an object instance of said first object type, wherein said object instance of said first

object type includes data from said second table associated with said first custom

attribute of said first object type. (See column 32, lines 55-59 "In step 1406, database

operations are supported by relating objects stored in the fixed mapped table to their

extensions in the flexible mapped table. The extensions are related to their respective

objects via the object ID field in each entry." The data associated with the first attribute

in the case is the object ID field.)

19.   Regarding claims 17 and 46, **Menon** additionally teaches said second table

stores values for custom attributes of a plurality of object types including said object

type, wherein the custom attributes are of said first data type (See FIG 11, item 1106a,

showing the custom attributes all of the String type – i.e. first data type.);

said third table stores values for custom attributes of a plurality of object types

including said object type, wherein the custom attributes are of said second data type

(See FIG 11, item 1106b, showing the custom attributes all of the Char type – i.e.

second data type);

and the method further comprises assigning to every instance of every object

type of said plurality of object types an instance identifier value that is unique relative to

every instance of every object type of said plurality of object types (See FIG 11, item

1103).

20.   Regarding claims 20 and 49 **Menon** teaches the step of retrieving, in response to

a request for an object instance of said first object type, the value of said first custom

attribute associated with said object instance (See column 20, lines 9 – 13 "Each box

depicted within the second column 1105 represents a particular attribute for the

associated asset [object]. Each attribute typically comprises three elements: (1) an

attribute name; (2) an attribute value; and (3) an attribute type."); by performing the

steps of:

determining the data type of said first custom attribute from said third catalog

column of a catalog-table row stored in said catalog table (See column 20, lines 9 – 13

"Each box depicted within the second column 1105 represents a particular attribute for

the associated asset [object]. Each attribute typically comprises three elements: (1) an

attribute name; (2) an attribute value; and (3) an attribute type." And lines 14-16 "The

type is selected from a predefined list of types. As will be described below, each

predefined type is associated with one of the separate metadata tables." The type is

also stored in the attribute field.); wherein:

data in said at least one first catalog column of said catalog-table row matches

data identifying said first object type (See FIG. 11, item 1114); and data in said at least

one second catalog column of said catalog-table row matches data identifying said first

custom attribute (See FIG. 11, item 1116);

based on the data type of said first custom attribute, determining the identity of

said second table (See column 20, lines 14-16 "As will be described below, each

predefined type is associated with one of the separate metadata tales." The second

table here represents the custom attribute tables.); and retrieving, from a value column

of a third-table row stored in said second table, the value of said first custom attribute

(See column 20, lines 11-13 "Each attribute typically comprises three elements: (1) an attribute name; (2) an attribute value; and (3) an attribute type."), wherein:

data uniquely identifying said object instance matches data in at least one instance-identifying column of said second-table row (See FIG. 11, item 1103); and

data identifying said first custom attribute matches data in at least one attribute-identifying column of said second-table row (See FIG. 11, item 1103, where the object id matches).

21.   Regarding claims 21 and 50, **Menon** additionally teaches the step of storing in said catalog table data defining a custom object type, separate from said object type, wherein the step of storing said custom object type includes the step of inserting a row into said catalog table (See column 30, lines 43-45 "In this embodiment, the extensions (e.g., object B' 1202b) are stored using asset table 1102 of the flexible mapped portion."), wherein said row includes: within said at least one first catalog column, data that identifies said custom object type (See column 20, lines 11-13, specifically (1) name); within said at least one second catalog column, data that identifies a custom attribute of said custom object type (See column 20, lines 1 and 2, specifically object id), and within said at least one third catalog column, data identifying a data type of said custom attribute that is identified in said at least one second catalog column (See column 20, lines 11-13, specifically (3) type).

22.   Regarding claim 25, **Menon** additionally teaches a fourth table, separate from

said first, second and third tables, storing data for a third custom attribute of said first

object type, wherein said third custom attribute is of a data type that is different than the

data type of said first custom attribute and said second custom attribute (See FIG. 11,

item 1106n, showing a fourth table, and representing that there could be any number of

tables depending on the number of data types that were defined.)

Claim Rejections - 35 USC § 103

23.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set
> forth in section 102 of this title, if the differences between the subject matter sought to be patented and
> the prior art are such that the subject matter as a whole would have been obvious at the time the
> invention was made to a person having ordinary skill in the art to which said subject matter pertains.
> Patentability shall not be negatived by the manner in which the invention was made.

24.   Claims 3 and 32 are rejected under 35 U.S.C. 103(a) as being unpatentable over

Menon as applied to claim 1 above, and further in view of **Becker et al.** (hereinafter

Becker, US 2004/0267744).

        Menon teaches retaining, in said third table, values for said first custom attribute

of said first object type and said second custom attribute of said second object type.

(See figure 11, item 1106 that stores the values of the custom attributes.)

        Menon fails to teach the step of upgrading said application, wherein upgrading

said application comprises the steps of: processing the data stored in said first table,

wherein processing comprises: creating a first replacement table to hold the data from

said first table; copying the data from said first table to said first replacement table, wherein data from said one or more default attributes of said first object type is copied from said first table into said first replacement table; and deleting said first table; processing the data stored in said second table, wherein processing comprises: creating a second replacement table to hold the data from said second table; copying the data from said second table to said second replacement table, wherein data from said one or more default attributes of said second object type is copied from said second table to said second replacement table; and deleting said second table.

However, **Becker** teaches the step of upgrading said application, wherein upgrading said application comprises the steps of: processing the data stored in said first table, wherein processing comprises: creating a first replacement table to hold the data from said first table (See page 1, paragraph 0013] "a destination table is created in the first database system, the destination table having a second data structure which is different than the first data structure."); copying the data from said first table to said first replacement table (See page 1, paragraph [0015] "the destination table is copied to a copy of the destination table in a second database system..."), wherein data from said one or more default attributes of said first object type is copied from said first table into said first replacement table (See page 1, paragraph [0015] "...the second data structure being retained in the copy of the destination table"); and deleting said first table (See page 1, paragraph [0019] "One option is the additional step of keeping the copy of the destination table, executed after the conversion step." Examiner interprets this to mean the standard would be to delete the table, if an option is to keep it.); processing the data

stored in said second table, wherein processing comprises: creating a second

replacement table to hold the data from said second table; copying the data from said

second table to said second replacement table, wherein data from said one or more

default attributes of said second object type is copied from said second table to said

second replacement table; and deleting said second table. (See page 4 paragraph

[0083] "The description allows a person skilled in the art to adapt the method to a

plurality of tables as well, for example by means of parallel processing." The above

claims are simply adapted to a second table as is anticipated in the reference.)

It would have been obvious to one with ordinary skill in the art to combine the

data storing method of **Menon** with the application upgrading of **Becker** because part of

the reason for separating the default attributes from the custom attributes is to make the

upgrading easier. Also, both **Menon** and **Becker** are operating on relational databases

with separated tables. It is for this reason that one of ordinary skill in the art would have

been motivated to include the step of upgrading said application, wherein upgrading

said application comprises the steps of: processing the data stored in said first table,

wherein processing comprises: creating a first replacement table to hold the data from

said first table; copying the data from said first table to said first replacement table,

wherein data from said one or more default attributes of said first object type is copied

from said first table into said first replacement table; and deleting said first table;

processing the data stored in said second table, wherein processing comprises:

creating a second replacement table to hold the data from said second table; copying

the data from said second table to said second replacement table, wherein data from

said one or more default attributes of said second object type is copied from said

second table to said second replacement table; and deleting said second table.

25.   Claims 15 and 44 are rejected under 35 U.S.C. 103(a) as being unpatentable

over **Menon** as applied to claim 12 above, and further in view of **Becker et al.**

(hereinafter **Becker**, US 2004/0267744).

Menon teaches retaining, in said second table, values for said first custom

attribute of said first object type and said second custom attribute of said second object

type and retaining, in said third table, values for said second custom attribute of said

object type. (See figure 11, items 1106a-n that store the values of the custom attributes

in separate tables, by object type.)

Menon fails to teach the step of upgrading said application, wherein upgrading

said application comprises the steps of: processing the data stored in said first table,

wherein processing comprises: creating a first replacement table to hold the data from

said first table; copying the data from said first table to said first replacement table,

wherein data from said one or more default attributes of said first object type is copied

from said first table into said first replacement table; and deleting said first table;

However, **Becker** teaches the step of upgrading said application, wherein

upgrading said application comprises the steps of: processing the data stored in said

first table, wherein processing comprises: creating a first replacement table to hold the

data from said first table (See page 1, paragraph 0013] "a destination table is created in

the first database system, the destination table having a second data structure which is

different than the first data structure."); copying the data from said first table to said first

replacement table (See page 1, paragraph [0015] "the destination table is copied to a

copy of the destination table in a second database system..."), wherein data from said

one or more default attributes of said first object type is copied from said first table into

said first replacement table (See page 1, paragraph [0015] "...the second data structure

being retained in the copy of the destination table"); and deleting said first table (See

page 1, paragraph [0019] "One option is the additional step of keeping the copy of the

destination table, executed after the conversion step." Examiner interprets this to mean

the standard would be to delete the table, if an option is to keep it.)

It would have been obvious to one with ordinary skill in the art to combine the

data storing method of **Menon** with the application upgrading of **Becker** because part of

· the reason for separating the default attributes from the custom attributes is to make the

upgrading easier. Also, both **Menon** and **Becker** are operating on relational databases

with separated tables. It is for this reason that one of ordinary skill in the art would have

been motivated to include the step of upgrading said application, wherein upgrading

said application comprises the steps of: processing the data stored in said first table,

wherein processing comprises: creating a first replacement table to hold the data from

said first table; copying the data from said first table to said first replacement table,

wherein data from said one or more default attributes of said first object type is copied

from said first table into said first replacement table; and deleting said first table.

Conclusion

26.   The prior art made of record and not relied upon is considered pertinent to

applicant's disclosure.

Millet et al. (2003/0154197) teaches separate custom value data tables.

Any inquiry concerning this communication or earlier communications from the

examiner should be directed to Dennis L. Vautrot whose telephone number is 571-272-

2184. The examiner can normally be reached on Monday-Friday 8:30-5:30.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's

supervisor, John Cottingham can be reached on 571-272-7079. The fax phone number

for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the

Patent Application Information Retrieval (PAIR) system. Status information for

published applications may be obtained from either Private PAIR or Public PAIR.

Status information for unpublished applications is available through Private PAIR only.

For more information about the PAIR system, see http://pair-direct.uspto.gov. Should

you have questions on access to the Private PAIR system, contact the Electronic

Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a

USPTO Customer Service Representative or access to the automated information

system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Dv
27 July 2006

JOHN COTTINGHAM
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100